Computational Thinking & Adaptieve Technologie

Eelco Braad | Hylke Faber

Introductie

Computational Thinking wordt door Stichting Leerplanontwikkeling (SLO) genoemd als een van de 21e - eeuwse vaardigheden; vaardigheden die voor elke leerling op de basisschool van belang zijn. De doelstelling van dit project is om inzicht te verkrijgen in het ontwerpen van en het werken met adaptieve toepassingen in het onderwijs. Hiervoor kijken we naar gepersonaliseerde Lego WeDo oefeningen waarmee computational thinking geleerd kan worden. In een evaluatie willen we met studenten in het lab ICT & Didactiek evalueren hoe dit bevalt.

Dit project kent drie hoofdbegrippen: computational thinking, Lego WeDo en adaptieve technologie. Wellicht is er in de literatuur of vanuit de praktijk al inzicht over deze begrippen. In dit overzicht geven we een korte indruk van de state-of-the-art op deze vlakken.

Computational Thinking

Verschillende instanties, waaronder Kennisnet en SLO zijn het erover eens dat computational thinking een belangrijke mentale vaardigheid is die op school onderwezen dient te worden. In haar oorspronkelijke uitspraak over computational thinking liet Jeannette Wing (2006) merken dat ze overtuigd was dat computational thinking net zo belangrijk zou zijn als schrijven, lezen en rekenen. Er zijn andere onderzoekers die eveneens in grote mate overtuigd zijn van de meerwaarde van computational thinking (Barr & Stephenson, 2011; Grgurina, Barendsen, Zwaneveld, van Veen, & Stoker, 2014; Grover & Pea, 2013; Kafai & Burke, 2013; Lu & Fletcher, 2009; Voogt, Fisser, Good, Mishra, & Yadav, 2015; Wing, 2006).

Ook kan computational thinking ook bijdragen aan *code-literacy*, het vermogen om computercode te lezen en schrijven, een vaardigheid die in de toekomst mogelijkerwijs steeds belangrijker zal worden (Román, 2016). Andere onderzoekers stellen dat het toekomstige bèta-onderwijs voor een groot deel een beroep zal doen op computational thinking denkvaardigheden (Henderson, Cortina, Hazzan, & Wing, 2007).

Uit onderzoek van Seymour Papert blijkt dat het leren omgaan met computers en het begrijpen van de manier waarop ze programmeerd worden kan bijdragen aan de metacognitie van kinderen (Papert, 1980). Een concreet voorbeeld hiervan is dat kinderen leren dat het maken van fouten een stap is in het proces van het komen tot een juiste oplossing. Papert legt de link met het verhelpen van programmeerfouten in een programma. Het verbeteren van deze fouten is een belangrijk onderdeel van het proces van programmeren. Door op deze manier naar fouten te kijken kunnen deze geïnterpreteerd worden als waardevolle leermomenten in plaats van situaties die vermeden moeten worden. Dit kan waardevol zijn voor kinderen die bang zijn voor het maken van fouten.

Tot slot kan computational thinking kinderen in staat stellen niet alleen informatica kunnen consumeren, maar ook te kunnen creëren en zich er bijvoorbeeld op een creatieve manier mee te kunnen uiten (Brennan & Resnick, 2012). Mogelijkerwijs zou computational thinking ook gebruikt kunnen worden ter

ondersteuning van de uitleg in complexe aspecten van andere schoolse vakken (Franklin et al., 2015; Grover & Pea, 2013).

Over de precieze definitie van computational thinking bestaat onenigheid. Verschillende onderzoekers en instanties omarmen vaak hun eigen definities, die in meer of mindere mate overeenstemming met elkaar vertonen. Ondanks deze onenigheid lijkt er wel overeenstemming te zijn onder onderzoekers dat computational thinking gezien kan worden als een uitbreiding van het probleemoplossend vermogen (Barr & Stephenson, 2011; Grover & Pea, 2013; International Society for Technology Education & Computer Sience Teachers Association, 2011; Lu & Fletcher, 2009; Wing, 2008).

Twee onderzoekers uit Engeland hebben een meta-analyse verricht waarbij er 39 studies vergeleken zijn, op die manier tot een gezamenlijke definitie te komen, gebaseerd op de overeenkomsten tussen deze 39 studies en onderzoekers (Selby & Woollard, 2014). Samenvattend kan computational thinking volgens Selby en Woollard als volgt omschreven worden: een set denkvaardigheden die het probleemoplossend vermogen versterken. Het toepassen van computational thinking omvat dat het probleem op één of meerdere niveaus van complexiteit wordt beschreven en aangepakt. computational thinking houdt tevens in dat het probleem in kleine deelproblemen opgedeeld wordt. Een mogelijke oplossing wordt vervolgens op een stapsgewijze manier beschreven en getoetst op haalbaarheid, effectiviteit en efficiëntie. Daarbij wordt ook gekeken naar hoe de oplossing in andere situaties toegepast kan worden en in hoeverre deze door een computer uitgevoerd kan worden.

Sinds 2015 heeft SLO een online publicatie uitgebracht een uitleg van computational thinking (SLO, 2015). Inhoudelijk vertoont deze publicatie veel gelijkenissen met de definitie van computational thinking zoals uitgegeven door de CSTA en ISTE, twee Amerikaanse onderwijsinstellingen (Barr & Stephenson, 2011). Hieronder volgt een korte opsomming van de elementen waaruit computational thinking volgens SLO bestaat:

1) <u>Gegevens verzamelen</u>
   Het bedenken welke gegevens relevant zullen zijn voor het vinden van een oplossing voor de probleemsituatie, en het verzamelen van deze gegevens uit verschillende bronnen.
2) <u>Gegevens analyseren</u>
   Het analyseren en ordenen van gegevens zodat er conclusies uit te getrokken kunnen worden, bijvoorbeeld door het interpreteren van grafische data of statistische methoden.
3) <u>Gegevens visualiseren</u>
   Het weergeven of representeren van data op een manier waarop deze gemakkelijker te interpreteren zijn, zoals het tekenen van grafieken en tabellen.
4) <u>Probleem decompositie</u>
   Het kunnen opdelen van een groot complex probleem in kleinere en overzichtelijke deelproblemen, waardoor het vinden van een oplossing voor het grotere probleem makkelijker wordt.
5) <u>Abstractie</u>
   Het maken van een versimpeling van de probleemsituatie, door het achterwege laten van niet relevante details.
6) <u>Algoritmes en procedures</u>

Het opstellen van een lijst korte en ondubbelzinnige instructies, waardoor een oplossing duidelijk gerepresenteerd wordt, alsmede het kunnen generaliseren van een algoritme naar soortgelijke probleemsituaties

7) Automatisering
   Het kunnen automatiseren van de uitvoering van het eerder opgestelde algoritme door een *computing agent*, zoals een compiler of een rekenmachine.
8) Simulatie en modellering
   Het kunnen beschrijven of weergeven van een complexe probleemsituatie in een model of simulatie, waarbij de complexiteit verminderd is om een duidelijker beeld van de situatie te schetsen.
9) Parallelization
   Het gelijktijdig kunnen laten uitvoeren van een algoritme door een *computing agent.*

Binnen deze negen aspecten van computational thinking bestaat enige overlap. Met name het concept van *abstractie* komt in veel aspecten terug. Onderzoekers stellen dat abstractie het aspect blijkt te zijn dat computational thinking onderscheidt van andere manieren van denken (Grover & Pea, 2013). Ook Jeanette Wing, die de term computational thinking introduceerde, bestempelde abstractie als het essentiële aspect van computational thinking (Wing, 2008). De vaardigheid van het kunnen abstraheren, het maken van een versimpeling van de probleemsituatie, lijkt in veel aspecten van computational thinking terug te komen. In eerste instantie is dit het geval bij het aspect *simulatie en modellering*. Een model of simulatie is het resultaat van een abstractie en kan misschien wel als bewijs dienen dat er een proces van abstractie aan vooraf is gegaan. *Probleem decompositie* kan gezien worden als een proces wat noodzakelijk is om tot een abstractie te komen. Het kunnen opdelen van een probleemsituatie in meerdere afhankelijke deelproblemen is een voorwaarde een correcte versimpeling van de situatie te creëren. Ook het opstellen van algoritmes om een probleem- of oplossingsstrategie te beschrijven draagt bij aan het proces van abstraheren.

Leren met LEGO

LEGO Education ontwikkelt verschillende materialen die speciaal bedoeld zijn voor het basisonderwijs. LEGO WeDo 2.0 is een van deze materialen. De set bestaat uit een aantal van de bekende LEGO blokjes en een hub. Deze hub is via een tablet of computer te programmeren. Op de hub kunnen motoren of verschillende sensoren aangesloten worden, die vervolgens weer gebruikt kunnen worden bij het programmeren van de robot.

De LEGO WeDo 2.0 set is ontwikkeld om programmeren en coderen te onderwijzen voor kinderen in groep 1 t/m 6 (LEGO Education, 2014b). Naast het ontwikkelen van lesmateriaal verricht LEGO Education ook onderzoek naar effectieve leermethoden. Zo heeft LEGO Education een model ontwikkeld waarbinnen leren op een vrij en exploratieve manier kan plaatsvinden. Hierbij staat 4 C's centraal: *connect*, *construct*, *contemplate* en *continue*. Dit model beschrijft hoe leren kan ontstaat bij het toepassen van LEGO lesmateriaal (LEGO Education, 2014a).

Connect: Leertaken van LEGO stellen de leerling in staat om zelf met ideeën en oplossingen te komen. Leerlingen worden uitgedaagd om vooraf vragen te stellen en de taak te ontdekken

Construct: Ook tijdens het bouwen van de bouwtuigen van LEGO vindt een leerproces plaats. Bovendien kan samenwerking dit proces verstevigen, omdat samenwerking vaak tot betere of uitgebreidere oplossingen leidt.

Contemplate: Het lesmateriaal van LEGO nodigt uit om de gevonden resultaten uitvoerig te evalueren en te bespreken.

Continue: Na elke taak begint een nieuwe (uitdagendere) taak die zodanig opgezet is dat de leerling zoveel mogelijk in een staat van *Flow* blijft.

Het hierboven beschreven model is bedoeld om leerlingen in een staat van *Flow* te krijgen. Dit komt tot stand wanneer de taak net moeilijk genoeg is en aansluit bij het huidige niveau van de leerling. Dit sluit aan bij de theorie van Vygotsky (1978) over de Zone van de Naaste Ontwikkeling, die ook de nadruk legt op de moeilijkheidsgraad van de taak. Dat niveau zou net iets hoger moeten zijn dan waar de leerling op dit moment alleen toe in staat is. Met hulp van buitenaf, door bijvoorbeeld een volwassene of door mede-leerlingen, kan de leerling het hogere niveau bereiken. LEGO Education onderschrijft ook de positieve invloed die samenwerking op het leerproces kan hebben.

Recente onderzoeken hebben uitgewezen dat het lesmateriaal van LEGO gebruikt kan worden om computational thinking te onderwijzen (Chetty, 2015; Mayerová & Veselovská, 2017; Pinto-llorente, Casillas-Martín, Cabezas-González, & García-Peñalvo, 2016).

Adaptieve Educatieve Technologie

Technologie wordt steeds gemakkelijker bruikbaar en aanpasbaar waardoor allerlei toepassingen hun weg hebben gevonden naar het onderwijs. Een klaslokaal zonder digibord of een studie zonder elektronische leeromgeving zijn ondenkbaar geworden – veel onderwijs wordt zelfs geheel digitaal en online aangeboden. Onder educatieve technologie verstaan we die toepassingen die ingericht zijn om een bepaald leerdoel te bereiken. Wanneer educatieve technologie zich dynamisch aanpast op de individuele gebruiker en haar omgeving spreken we van adaptieve educatieve technologie (Brusilovsky, 2001).

Een goed voorbeeld is DuoLingo: een platform waarin je verschillende talen kunt leren. Elke taal is georganiseerd in blokjes stof; bijvoorbeeld basis blokken en blokken rondom verschillende thema's zoals voedsel of familie. Binnen elk blokje leer je nieuwe woorden en zinsconstructies aan de hand van geschreven en gesproken oefeningen. Welke blokjes beschikbaar zijn of aangeraden worden hangt af van je voorgaande prestaties en tempo. Het systeem past zich dus aan de gebruiker aan om motivatie en leereffect te ondersteunen (Kerr, 2014).

De belangrijkste ontwerpkeuzes voor een adaptief systeem zijn: (1) wat meet het systeem van de gebruiker (input), (2) hoe probeert het systeem te adapteren (output). Dit hangt nauw samen met (3) welk doel wordt geprobeerd te bereiken (goal). Qua input kan de complexiteit variëren van directe metingen zoals bijvoorbeeld aantal muisklikken tot persoonlijkheidseigenschappen van de gebruiker die worden ingeschat. Qua output is er een onderscheid te maken tussen de inhoud en het proces: inhoud kan verschillend gepresenteerd worden of al dan niet aangeboden voor een bepaalde gebruiker; in het proces zijn variabelen zoals tempo, timing, moeilijkheidsgraad en herhaling gemakkelijk aan te sturen. Qua goals wordt vaak gericht op een hoger leereffect in samenhang met een hogere motivatie (Kerr, 2014; Wannemacker, Clarebout, & Causmaecker, 2010). Dit is met name terug te zien in de vele adaptive learning toepassingen die samenvallen met het toepassen van gamification of serious games (Burgos et

al., 2007; Law & Sun, 2012; Peirce, Conlan, & Wade, 2008; Torrente, Moreno-Ger, & Fernández-Manjón, 2008).

Er zijn verschillende manieren om van input tot output te komen: het beslismechanisme van het adaptieve systeem. Veel systemen volgen een data-driven approach waarbij van een grote groep gebruikers veel data wordt verzameld, waarna door clustering wordt geprobeerd om een nieuwe gebruiker in een bepaalde groep in te delen. Het systeem is daarmee dus een soort black box – het werkt volgens het meerderheidsprincipe maar er zit geen specifieke kennis of leerlingen of het leerproces in. Daar tegeonver staan systemen met een meer theory-driven approach waarbij vantevoren wordt bedacht wat een goed werkzaam mechanisme kan zijn. Een voorbeeld hier van is een knowledge space tree waarbij de leerstof wordt verdeeld in kleinere onderdelen en leerdoelen, die vervolgens in een boomstructuur aan elkaar gekoppeld zijn (Sitthisak & Gilbert, 2013)

Praktijk

Er zijn in de onderwijspraktijk al diverse voorbeelden van adaptieve technologie te vinden. Voor het basisonderwijs is Rekentuin (https://www.rekentuin.nl/) een online omgeving waarin geoefend kan worden met rekenen. Rekentuin is adaptief: het past de moeilijkheidsgraad aan op het niveau van de speler ("Adaptief: alle spelers oefenen binnen Rekentuin altijd automatisch op hun eigen niveau...").

Knewton (https://www.knewton.com/) is een omgeving gericht op het hoger onderwijs en gebruikt een data-analytics aanpak om de juiste inhoud op het juiste moment aan een student aan te bieden. Dit gaat dus zowel over moeilijkheidsgraad als leerpad en tempo.

Daarnaast is er ook kritiek: met name op het wegcijferen van de mens in het onderwijs en over de ethiek van grote bedrijven die de educatieve wereld dreigen te overheersen (Kerr, 2014).

Referenties

Barr, B. V., & Stephenson, C. (2011). Computational Thinking to K-12 : What is Involved and What is the Role of the Computer Science Education Community ?, *2*(1), 48–54.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *Annual American Educational Research Association Meeting, Vancouver, BC, Canada*, 1–25.

Brusilovsky, P. (2001). Adaptive hypermedia. *User Modeling and User-Adapted Interaction*, *11*(1–2), 87–110. http://doi.org/10.1023/A:1011143116306

Burgos, D., Moreno-Ger, P., Sierra, J. L., Fernandez-Manjon, B., Specht, M., & Koper, R. (2007). Building adaptive game-based learning resources: The integration of IMS Learning Design and <e-Adventure>. *Simulation & Gaming*, *39*, 414–431. http://doi.org/10.1177/1046878108319595

Chetty, J. (2015). The Notion of Lego© Mindstorms as a Powerful Pedagogical Tool: Scaffolding Learners through Computational Thinking and Computer Programming. *The Independent Journal of Teaching and Learning*, *10*, 69–84.

Franklin, D., Hill, C., Dwyer, H., Iveland, A., Killian, A., & Harlow, D. (2015). Getting Started in Teaching and Researching Computer Science in the Elementary Classroom. *Proceedings of the 46th ACM Technical Symposium on Computer Science Education - SIGCSE '15*, 552–557. http://doi.org/10.1145/2676723.2677288

Grgurina, N., Barendsen, E., Zwaneveld, B., van Veen, K., & Stoker, I. (2014). Computational Thinking Skills in Dutch Secondary Education. *Proceedings of the 14th Koli Calling International Conference on Computing Education Research*, 173–174. http://doi.org/10.1145/2674683.2674704

Grover, S., & Pea, R. D. (2013). Computational Thinking in K-12: A Review of the State of the Field. *Educational Researcher*, *42*(1), 38–43. http://doi.org/10.3102/0013189X12463051

Henderson, P. B., Cortina, T. J., Hazzan, O., & Wing, J. M. (2007). Computational thinking. *Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education*, 195–196. http://doi.org/10.1145/1227310.1227378

International Society for Technology Education, & Computer Sience Teachers Association. (2011). *Operational Definition of Computational Thinking for K-12 Education*.

Kafai, Y. B., & Burke, Q. (2013). Computer Programming Goes Back To School. *Phi Delta Kappan*, *95*(1), 61–65.

Kerr, P. (2014). Adaptive Learning. *CritiquEd*, *1*.

Law, E. L. C., & Sun, X. (2012). Evaluating user experience of adaptive digital educational games with Activity Theory. *International Journal of Human Computer Studies*, *70*(7), 478–497. http://doi.org/10.1016/j.ijhcs.2012.01.007

LEGO Education. (2014a). A System for Learning.

LEGO Education. (2014b). Computing Scheme of Work.

Lu, J. J., & Fletcher, G. H. L. (2009). Thinking about computational thinking. *ACM SIGCSE Bulletin*, *41*(1), 260. http://doi.org/10.1145/1539024.1508959

Mayerová, K., & Veselovská, M. (2017). How to Teach with LEGOWeDo at Primary School. In M. Merdan, W. Lepuschitz, G. Koppensteiner, & R. Balogh (Eds.), *Robotics in Education, Advances in Intelligent Systems and Computing* (pp. 55–62). Cham: Springer. http://doi.org/10.1007/978-3-319-42975-5

Papert, S. A. (1980). *Mindstorms: Children, Computers, and Powerful Ideas*. BasicBooks. http://doi.org/10.1007/s007690000247

Peirce, N., Conlan, O., & Wade, V. (2008). Adaptive educational games: Providing non-invasive personalised learning experiences. *Proceedings - 2nd IEEE International Conference on Digital Game and Intelligent Toy Enhanced Learning, DIGITEL 2008*, 28–35. http://doi.org/10.1109/DIGITEL.2008.30

Pinto-llorente, A. M., Casillas-Martín, S., Cabezas-González, M., & García-Peñalvo, F. J. (2016). Developing Computational Thinking via the Visual Programming Tool: Lego Education WeDo. In F. J. García-Peñalvo (Ed.), *Proceedings of the Fourth International Conference on Technological Ecosystems for Enhancing Multiculturality (TEEM'16)* (pp. 45–50). New York, NY: ACM.

Román, M. (2016). Computational Thinking Test: Design Guidelines and Content Validation. *Proceedings of EDULEARN15 Conference 6th-8th July 2015, Barcelona, Spain*, (February).

Selby, C. C., & Woollard, J. (2014). *Refining an Understanding of Computational Thinking*.

Sitthisak, O., & Gilbert, L. (2013). Integrating Competence Models with Knowledge Space Theory for Assessment, 1–8.

SLO. (2015). Een voorbeeldmatig leerplankader.

Torrente, J., Moreno-Ger, P., & Fernández-Manjón, B. (2008). Learning Models for the Integration of Adaptive Educational Games in Virtual Learning Environments. *Learning*, *5093*, 463–474. http://doi.org/10.1007/978-3-540-69736-7_50

Voogt, J., Fisser, P., Good, J., Mishra, P., & Yadav, A. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. *Education and Information Technologies*, 715–728. http://doi.org/10.1007/s10639-015-9412-6

Vygotsky, L. S. (1978). *Mind in Society: The Development of Higher Psychological Processes*. (M. Cole, V. John-Steiner, S. Scribner, & E. Souberman, Eds.). Cambridge, Massachusetts; London, England: Harvard University Press. http://doi.org/10.1007/978-3-540-92784-6

Wannemacker, S. de, Clarebout, G., & Causmaecker, P. (2010). *Interdisciplinary Approaches to Adaptive Learning*. Springer.

Wing, J. M. (2006). Computational Thinking. *Communications of the ACM*, *49*(3), 33–35. http://doi.org/10.1145/1118178.1118215

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions. Series A, Mathematical, Physical, and Engineering Sciences*, *366*(1881), 3717–3725. http://doi.org/10.1098/rsta.2008.0118